

**Version 1.1**

**Date: 2018/12/09**

Abstract

**The objective of this document is to describe the Catalogue Maintenance application that supports structures and applications driving the sales decision support business requirements**

Catalogue Maintenance

*Define All Structures and Application supporting lookup demands*

**Table of Contents**

Document approval and distribution list 2

1. Introduction 3

2. Audience 4

3. Objectives 5

4. Database Design Objectives 5

5. Data normalisation 8

6. Multiple suppliers for same part 9

7. Alternates, supersessions and replacement parts 9

8. Catalogue source material 10

9. Catalogue maintenance work flow 10

10. Supplier Pricing 11

11. Currency changes 11

12. Base cost calculations – optimisation 12

13. Recall of faulty / safety risk goods sold and on hand 12

14. Kitting and service sets 12

15. Supplier part profiles 13

16. Stocking Item (SKU) Maintenance 13

16.1 Group Code maintenance 13

16.2 Base Item maintenance 16

16.3 Catalogue – Bill of material node maintenance 21

16.4 Stock Mark-up workbench 25

16.5 Catalogue Supplier Price Maintenance 28

17. Dependencies 28

18. Database entities and relationships 30

19. Programs 32

19.1 MS Windows Executables 32

19.2 SQL Stored Procedures 32

20. Acceptance 36

# Document approval and distribution list

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Name / Title** | **Signature** | **Date** |
| **Document Type / purpose** | | | |
| Prepared by | Graham Smith Jaco van der Westhuizen |  |  |
| Reviewed by |  |  |  |
| Approved by |  |  |  |

# Introduction

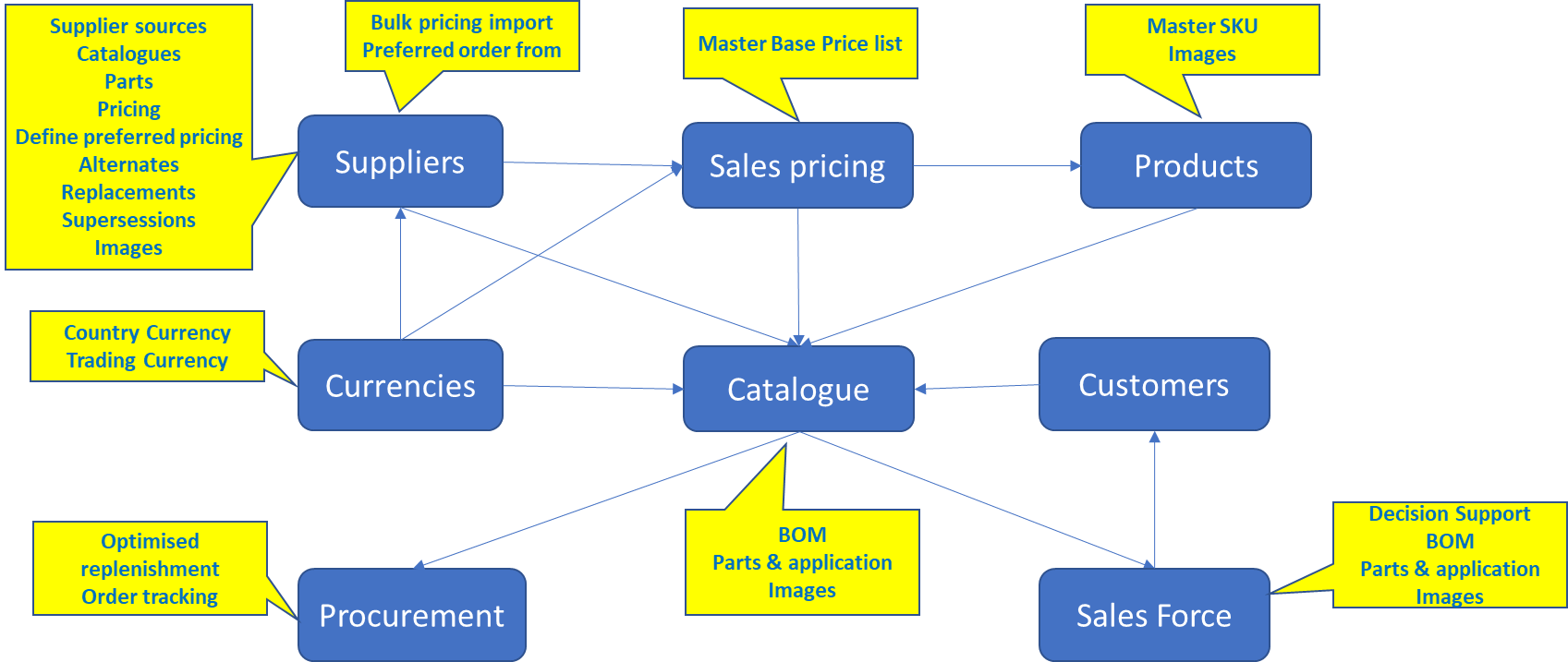
The so-called **Catalogue Lookup**represents an end user tool that is able to promote sales of parts to the industry through systems-based intelligence that is created using data captured in various sub systems in ePart.

Systems functionality exists to consolidate the data in contained in the various subsystem in an optimised manner in support of business requirements.

Additionally, there is functionality to create assemblies and sub-assemblies that represent the construct of parts and their relevant applications.

The following diagram provides a high-level end to end view of the ePart Catalogue system with some indicative functionality guideline annotations relative to sub-system functionality.

**Diagram 1.1: *A high level view of the ePart catalogue sub-system integration***



There is no ***redundant data in ePart due to the end to end systems and application integration***, consequently when reviewing the ePart ***catalogue*** end to end, it may lead to a misconception that it is complex. However, by brining all the sub-system components together in a unified manner it exposes the inter sub-system dependencies properly and negates the need to re-capture the same data repeatedly.

About the ***catalogue*** maintenance section, there are a number of functions that ease the repetitive work of catalogue engineers; some of these are:

* Creating tree structures of assemblies and sub-assemblies in line with the manufacturing industry and resembles the motor manufacturing industry method
* Creating assembly templates that can be re-used within the assembly tree
* Inheritance of template assemblies onto parent assemblies with or without future change inheritance.
* Example of this is the Ford 1000cc engine used in more than one vehicle. The engine assembly is created as a template and then linked onto the various vehicles where it is used. After linking, the linked version can be changed for subtle variances specific to the linked to vehicle. Although the example is for vehicle and engine, the same principal can be used at any level within the tree structure.
* Assemblies may be SKU items that can be sold i.e. turbo charger. However, a turbocharger may have sub-assemblies that could be selling SKU’s as well i.e. shaft bearings. Thus, the tree allows for the opportunity to define sellable SKU items as well as the constituent components that make up the SKU assembly.
* It must be noted that one or more images can be added at SKU level and at assembly level in the tree. The assembly level image would be applicable where an image would assist the selling process for the question ‘Does the bearing fit this gearbox?’ and we do not stock gearboxes.

# Audience

Sales

Cataloguing

Management

# Objectives

To optimise the definition process of creating structures and relationships that depict the ***real-world*** of stocking items and their related applications. In support of this, the database structures are defined using an ***inverted tree structure to manage the so-called bill of material definition.***

Follow the real-world parts identification allowing these to resolve back to the ePart base number (SKU)

Provide parts and assembly images to assist in the identification process, both for purchasing and sales teams.

Include as much detail as is possible so that operationally the business is optimised through the availability of pertinent information at the time. Illustratively, tariff heading in support of imported goods as well as for the export to neighbouring countries.

Have relevant dashboards through which the underlying data can be observed, and management decision made to improve business processes i.e:

* Selecting best supplier option for replenishment
* Define which supplier pricing to be used when calculating standard selling price
* Define which supplier pricing to be used when calculating wholesale price lists
* View sales trends to optimise pricing overall

# Database Design Objectives

Analysis of the data available, data gathering from various sources, relationships of parts to applications, it became evident that a ***formalised hierarchical structure*** (inverted tree) would be important in the data persistence implementation to support operational functionality.

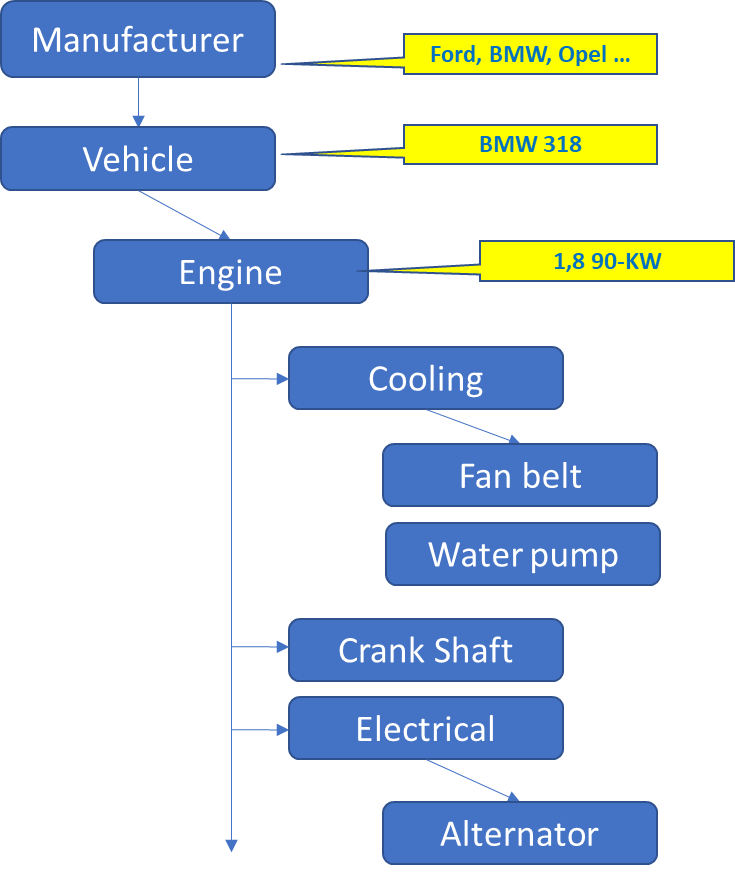
Contained in the legacy (COBOL based) catalogue that was to be replaced, was details of some part to application details. Although simplistic in representation, there was enough textual detail in the parts description to infer an inverted tree representation which was thought to be a suitable fit for the data available.

Some of the historical experience in the developer team was to develop solutions with open ended parent child relationships which does solve the visual view of the parts to application relationships not unlike a computer system inverted tree of directories and sub-directories i.e. Windows.

The following image provides a high-level view of how it would apply in the ePart catalogue system.

Inherently, the design is the inverted tree system is totally agnostic in terms of the specific assemblies and related sub-assemblies and parts linked. The implementation is quite capable of defining a food menu (pizza anyone) and link the recipe and ingredients to make a specific pizza on a menu!

This then allows the organisation the opportunity to create many bills of material representative of specific applications and their underlying catalogue parts.

**Diagram 4.1: *Provides a high-level view of the hierarchical data representation needed for parts to application relationships***

The following simplified diagram illustrates the database implementation that supports the inverted tree BOM implementation with linking to parts (SKU) as part of the leaf nodes:

**Diagram 4.2: *Inverted tree with leaf node parts linking***

This ***self-referencing*** relationship provides for unrestricted levels of relationships to be assembled in an ***assembly*** to ***Sub-Assembly*** without limits. The ***Part(s)*** relationship provides the ***item level component stocking attributes*** as depicted in the following diagram:

**Diagram 4.3 provides a view of how the relationship is used to depict an easily viewed construct**

The assembly / sub-assembly explosion indicates how elements with related attributes can be viewed with the highlighted entry indicating the actual stock item.

By design the solution provides for the linking of assemblies to stocking items even when the stock item is made up of sub-assemblies i.e. a turbo charger can be sold as a complete unit yet some parts making up the turbo charger can be sold separately.

To add a level of governance, there are a number of system and application defined rules to prevent end-users from making obvious mistakes i.e. linking a vehicle to an engine or linking a turbocharger to a turbocharger to turbo shaft bearing.

Additionally, the application make provision for the creation of ***template*** assemblies (usually done by senior staff with experience) that are then used by cataloguers through ***inheritance.*** This means that when declaring a new vehicle application, the work does not need to be repeated – use the appropriate template.

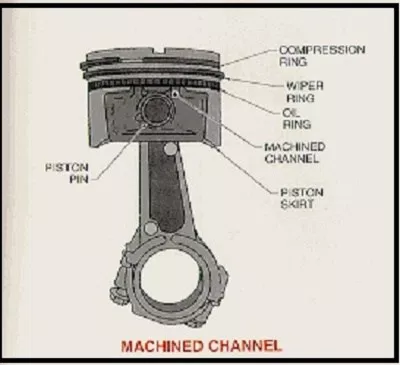
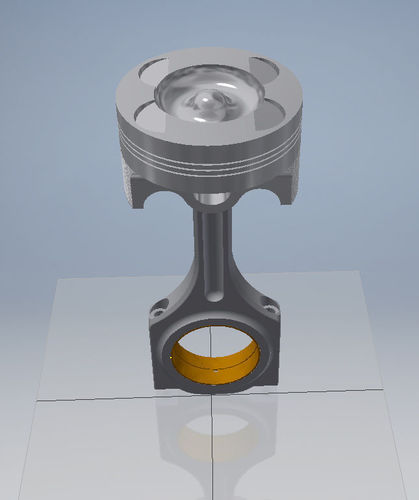
***Note that there are more control attributes such as assembly type, code and others. Refer to the full entity relationship chart for more details***

# Data normalisation

When the database design was undertaken, the understanding was gained that the traditional data normalisation (5th normalisation) against each product type could not succeed against the variety of SKU types in existence. This in support when searching for a suitable bearing size vs a piston of a type.

By way of an example, a piston as a type has properties that are generally repeatable between pistons of different piston applications i.e. attributes types of the following:

* Diameter
* Skirt length
* Compression rings
* Wiper rings
* Oil rings
* Gudgeon pin size
* …

On the other hand, ball bearings also have repeatable attributes between ball bearing types with attributes as follows that do not match those of a piston:

Speed Ratings

Reference speed: 75000 r/min

Limiting speed: 48000 r/min

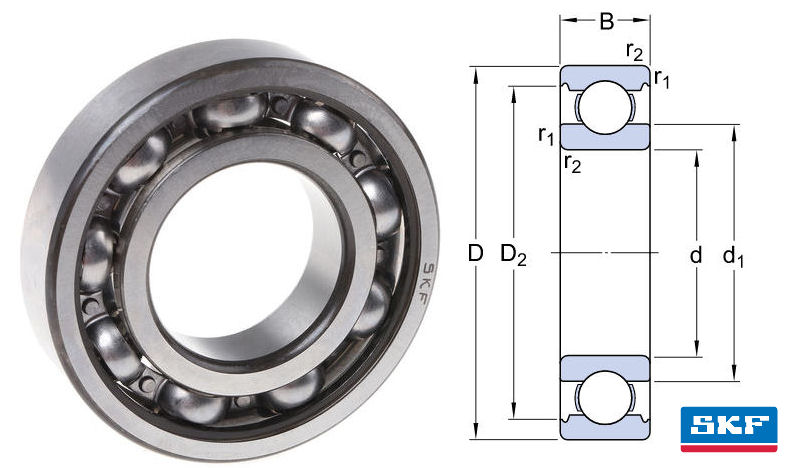
Dimensions mm

d1: 12.1

D1: 17.6

D2: 19.2

r1,2 min: 0.3



From a technical perspective the vast number of SKU (base numbers / parts) make a standard data normalisation approach an almost impossible task.

The design team of the day opted to allow the SKU properties to be user definable where a SKU is assigned a type and one or more properties against the type defined. Consequently, the data capturing program is SKU type aware and will limit the user input to the defined properties. This removed the need to perform data normalisation on a formalised manner but retained the ability to perform lookup searching against the defined attributes as well

# Multiple suppliers for same part

As part of the catalogue maintenance sub-system is to work through parts catalogues received from various manufacturers / suppliers. Details pertaining to the items listed in the catalogue are captured or updated without necessarily linking said part number to a system SKU (base number). It is assumed that at least that if there is a reference it will trigger a need to do further research to identify exactly the part and its application.

If at any time the un-linked part is identified against a SKU (base number) or another supplier part reference which is linked, the un-linked item is automatically cascade linked using the qualified detail exposed.

This feature allows purchasing staff to identify alternate suppliers from whom to order from.

It is a reality that OEM allocate part identifications per their Design & Manufacturing process that is not used by replacement part manufacturers – they use their own part identification system. However, the alternate part manufacturers do provide OEM referencing.

In support of this, ePart catalogue maintenance allows the creation of alternate identification part numbers from various suppliers / manufacturers. These linked back to the ePart Base number (SKU). Should a query be launched using the catalogued industry part reference, the ePart Base number can be resolved and hopefully the business concluded with success.

In the case of non-local suppliers (import), enough detail can be captured to allow for the export documentation to be completed i.e. tariff headings amongst others. Additionally, the product industry application detail (agricultural, commercial etc) needs to be maintained for government purposes so that import duties can be accounted for.

As part of the captured data, supplier parts profile is catered for i.e. dims, weight etc. Also, and of importance is the opportunity to capture the container bar code reference assigned to the supplier part. This bar code is linked to the SKU (Base Number) and can result in eliminating the print requirement for the ePart barcode.

Supplier bar codes can be captured as well allowing for speedier receiving

(Note: a Scoping document has been produced and needs management approval to execute on)

# Alternates, supersessions and replacement parts

As the assigned engineering skilled staff review their catalogue data, provision is made to mark parts in the following manner:

* Alternates is where a part is not the same but is otherwise a good alternate to use i.e. fan belts may have differing lengths but can do the same work.
* Supersessions is where a part is superseded for some reason by a newer part, it may very well be that the both the superseded and supersession are still serviceable. However, in the ePart system there is a specific flag setting that indicates that the superseded part may not be sold and will stop a sales transaction.

# Catalogue source material

Due to the large volume of source material received from manufacturers and distributors worldwide, it was a complex task to manage the material processed and not processed.

To ensure proper tracking, the catalogue source material is classified as if it were a public library. Each item (book or CD / DVD) was logged with various properties such as source of material, date received, number of pages etc into the ePart system.

Additionally, much of the material of late resides on the internet where paper based and to an extent CD/DVD are much reduced. To cater for these, the current system still applies in that the Web URL is captured with related properties.

Various dashboards / reports are available to reflect the status of each item in terms of date received and related progress

When reflecting of the ***lost sales*** due to part references not in record, the system is able to provide this kind of statistic and a manual process followed to allow the catalogue management to focus on these kinds of lost sales indicators.

# Catalogue maintenance work flow

To ensure that catalogue processing of source material moves forward against a given expectation, pages processed against the library total pages provides management with such statistics.

Paper based catalogues, based on management selection, are sent to printing companies for guillotining where the spine of the catalogues is cut releasing the catalogue pages. These loose pages are then passed through a high-speed duplex scanner for processing.

The scanned images are related to the reference data in the catalogue library and stored referentially. This means that the images are not stored in the database but externally with the database containing the location where the images can be found.

***Developer note***: If the images were stored in the database this would have a significant increase in the database size with no operational benefit. In fact, it would detract from the time to backup and especially time to recover in case of a database failure.

Each catalogue team member ***books*** an item out of the library and records each page processed. In the event where a paper-based book is selected, the scanned images are used directly off the computer screen.

At appropriate times, the cataloguer can select images on a specific page being processed by identifying the area and location of the image. Again the database does not contain the image at all; only the catalogue reference, page number, image co-ordinates and size is recorded in the database. Later when the image is requested, the page is retrieved and the image ***cut*** from the page image using the stores reference, co-ordinates and size. The selected are is the only part transmitted through the network. It is a proven benefit given the wide area network through which our customers operate from.

Pages are processed using text-based capturing (always) and where needed also adds product images as per the catalogue.

In the case of CD/DVD, these are pre-processed on a priority base as defined by management. Each ***PAGE*** is selected and recorded on ePart. This is a lot more time consuming than paper-based catalogues due to the inability to process the CD/DVD data.

For the internet-based catalogues the same scenario exists where the page based extraction does not exist.

Considerable effort was made to get suppliers to drop to ePart files that are page based as if scanned without ANY success.

A further investment was made to try and extract the individual pages using PDF reader software with limited success.

Similarly, the web-based extraction has yet to succeed. However, it would be good governance to see if better tools may be available to extract web-pages better. Some investment towards this is a business requirement.

# Supplier Pricing

Generally, supplier pricing is received in bulk format with periodic adjustment post receipt of bulk pricing.

For the bulk pricing there is a efficient process of pricing updaters importation with related error flagging where exceptions may be found. Generally, exceptions are around the current pricing model in ePart that calculate to an invalid sales price basis, usually under cost or too small a margin.

Similarly, the occasional price updates can be captured through a user interface with appropriate error reporting.

Post supplier price update, the active suppliers and related pricing is reviewed, and decisions made as to which supplier may be the flagged preferred supplier. Also, there is a flag that is set to define which supplier will be the preferred selling price basis to calculate from, this includes the defining of the ePart wholesale selling price.

To note, the pricing is always expressed in ZAR from the sales perspective even though the suppliers may be using their own currency for creditors settlement.

# Currency changes

Due to the fact the Engineparts has the ability to order stocking items internationally, up to date currency conversion factors are a reality.

The ePart system has been optimised to apply changed conversion factors on the fly and is able to apply such updates in less than 2 seconds whereas in the past (pew ePart) this would have consumed several weeks of preparation and processing to recalculate the selling pricing.

Provision has been made but not implemented, to allow standard costing apportionment to GL for cost incurred beyond the currency factors for attributes such as forward cover, insurance etc.

# Base cost calculations – optimisation

To optimise the selling price determination, the base price of selling price is pre-calculated to the point where customer profile determines the final selling price.

This need to re-calculate can be initiated from several sources i.e. change in preferred supplier, supplier price change, currency change, rules in base price determination etc.

To ensure that there is a common re-calculation routing an event log table with a trigger was implemented. Each price change event is logged, and the trigger fired with virtual real-time results.

# Recall of faulty / safety risk goods sold and on hand

Provision is made to identify parts with manufacturing / factory faults that cannot be sold. Additionally, some of said parts need to be recalled from end-users where possible.

Internally to ePart, these are identified as non-saleable and moved to a designated bin, ready for administrative intervention.

Intervention could be as little as following a write-off process with destruction through to awaiting supplier inspection.

In some instances, the supplier may replace the faulty parts or pass a credit for the faulty parts in which case the ePart claims process can be followed.

ePart is equipped to identify the customers to whom the faulty parts have been supplied to. The technical department is provided with a list of customer invoices for follow-up with.

Customers are required to return faulty goods. On receipt, Engineparts staff are required to bin returned goods to designated bin awaiting final inspection.

Any faulty parts sold by Engineparts customers to their customers is outside of this document scope and should be dealt with at an administrative level.

# Kitting and service sets

ePart provides functionality to create kitting of parts used to assist sales staff in selling more to customers (building a good customer experience).

By way of an example is where a customer requires all the parts for a service on a specific vehicle. Once the vehicle is identified, a specific group / identifying code presents the sales person with a list of potential parts that the customer should require for a ***service*** to be done.

The sales person can the offer the identified parts to the customer as a complete set – good customer experience

# Supplier part profiles

ePart allows for the capturing of supplier details pertaining to the item in how it would be supplied.

The objective of this functionality is to optimise other elements in the ePart systems. Examples of the type of data that can be captured is as follows:

* 1. Dimensions (being height, width and length)

The intent is to use this to measure the quantity of said part that can fir an appropriate bin.

* 1. Weight

The intent is to control the courier charges and exception management

* 1. Fragility

Headlamp lighting if dropped glass will break and as such will allow for pickers and packers to treat said product with caution.

* 1. Supplier bar code

The intent of the courier barcode as an alternate reference to the ePart SKU (base number). The application intent is to use the supplier barcode reference to pick, check & pack. This will eliminate the need to produce in-house labels as is the current practice.

# Stocking Item (SKU) Maintenance

## Group Code maintenance

Although this document refers to the catalogue maintenance, it includes the traditional SKU (stocking item maintenance) sub system as it forms integral part to the products that Engineparts manages and markets to its customers.

Due to the traditional representation of SKU identification of the Engineparts legacy system, this was retained.

The SKU identity is known as the ***Base Number*** within the ePart as retained from the legacy system

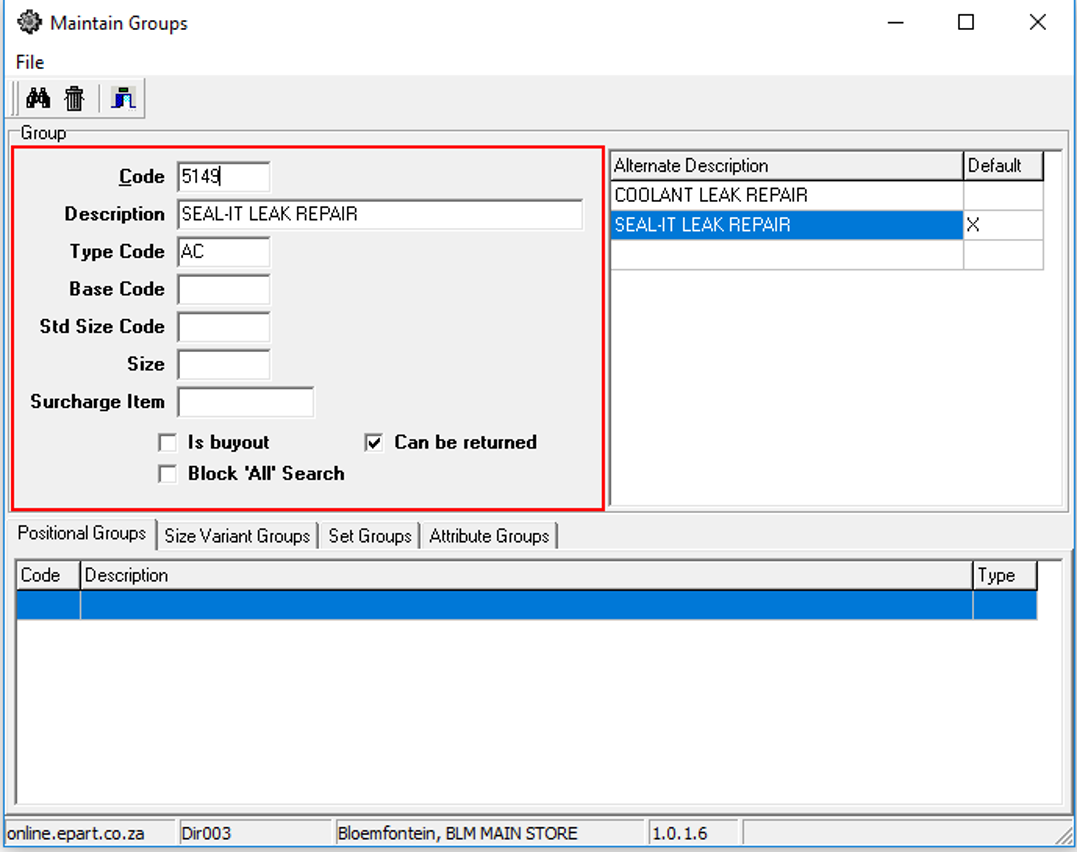
As per the legacy system, the SKU / Base Number is a singular value but in practice is composed of 2 sections as follows:

|  |  |
| --- | --- |
| Group code | 4 numeric characters and serves as a grouping i.e. 3500 = all water pumps |
| Serial number | 4 numeric characters representing a specific item i.e. 0110 = a specific vehicle application |

The Group Code specifically is managed by senior cataloguing staff once a decision has been formulated by the procurement policy team.

The following image represents the maintenance screen for group codes:

***Image 16.1 is the Group Code maintenance user interface***



The following is a guide to interpreting the specific group code profile fields that can be maintained:

* ***Type Code*** is a search key by product area of application and in this case, represents Coolant be it engine or air conditioning, radiator etc. The intent is to assist searching by sales staff / on-line customers within a generic category
* ***Std Size Code*** is the Base Item code for a standard size. This allows the serial number to be used for a specific part application and the group code is used to define the specific over / under size.

An example of this is where a piston ring application could be as follows:

In the case of a piston, the engine could be bored to a bigger size

|  |  |  |  |
| --- | --- | --- | --- |
| Serial number | Application | Group Code | Size |
| 0043 | Ford Echo Sport - piston | 9310 | Standard |
| 0043 | Ford Echo Sport – piston | 9311 | 10 thou bigger |
| 0043 | Ford Echo Sport – piston | 9312 | 20 thou bigger |

In the case of a crankshaft, the bearing faces can be milled and causes the bearing face to be smaller / under sized:

|  |  |  |  |
| --- | --- | --- | --- |
| Serial number | Application | Group Code | Size |
| 0061 | Ford Echo Sport - crankshaft | 7310 | Standard |
| 0061 | Ford Echo Sport | 7311 | 10 thou smaller |
| 0061 | Ford Echo Sport | 7312 | 20 thou smaller |

Although the numbering convention is somewhat odd from traditional numbering schemes, this is still a well thought through methodology in that an interested party can work from a serial number to get to the various over / under sizes easily.

* The size field represent the over / under size than can be searched vie the catalogue lookup functionality
* Surcharge item – this field allows for a surcharge to be raised in the case of an exchange of the replacement part i.e. batteries where the old battery core is expected as part of the transaction. If not raise a surcharge.

This also applied for turbo charger cores

* Is Buyout – not used
* Can be returned. These are items that will block when returned by customers.
* Block ‘ALL’ Search. This prevents searching across ALL serial numbers in a group code forcing the person with a interest from receiving a list of parts beyond his or he ability to interpret. This forces the user to be more specific in the search criteria to return a smaller set of results.

Note that the user interface has a number of tabs below the standard information and are used as follows:

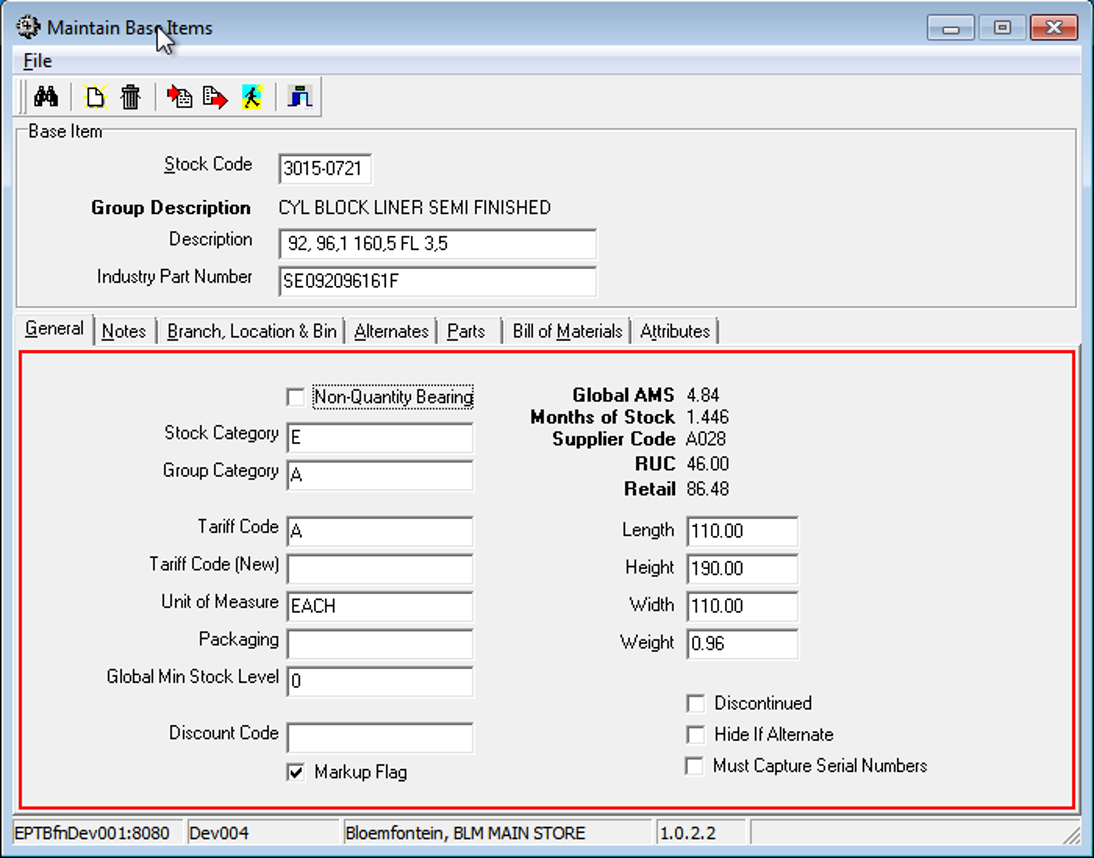
* Positional groups. In the same manner as under / over sizes are identified using a group code, the position where the specific item can be used is identified. By example a roller bearing may be used on the front left wheel assembly as well as the front right wheel assembly. This is particularly useful to identify that the same part can be used in a number of positions in an application.
* Size variant Group is a list of group codes with their variations in sizes. This implies that the serial numbers and group code do not have to have any conformance to identification characters
* Set Groups. This is used to identify a set of parts that could be used to assist sales and customers alike, when purchasing parts needed to i.e. service a specific vehicle. It proposes to the customer all the items that would be needed for this action.
* Attribute groups. This is a specialised form of descriptive assistance that indicates how the part should be fitted i.e. torque wrench value etc.

## Base Item maintenance

As indicated, the SKU / base number if used to identify a specific item code and is made up of 2 parts being the group code and serial numbers.

The following image provide a view of the base item maintenance user interface:

***Image 16.2 is the user interface to maintain base item details:***



The standard descriptive elements are as follows:

* Group Description. This is derived from the Group Maintenance user interface and recalled for display base don the 4 character that makes up the base number. Note that this is not an editable field
* Description. This is the description assigned to the serial number part of the SKU. In the example the group description is for a valve guide in a specific size group of valves whilst the specific value attributes are embedded into the description.
* Industry part number. This is the common, probably the OEM, number that most if not all manufacturers / sellers will be familiar with. The objective of this is to assist interested parties in identifying the Engineparts equivalent

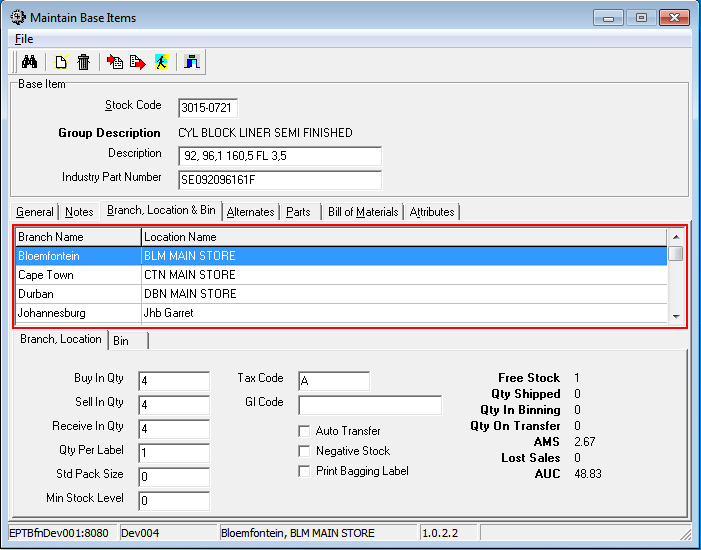
Note that there is a set of UI tabs that expand on the base item profile and described as follows:

**General tab** – this part extends the properties of each SKU / base item.

To note is that it is possible to define ***standard*** physical properties. However, ePart provides for supplier properties to be captured which as well should override the standard properties

* **Notes tab** – this allows authorised maintainers the opportunity to capture notation type data
* **Branch, location & bin.** The detail provides the maintainer the ability to profile SKU with more selective detail.

The ePart system provides for multiple locations per Company as evidenced by the view below. Note, that multi-company is possible as implemented with Namibia as well.



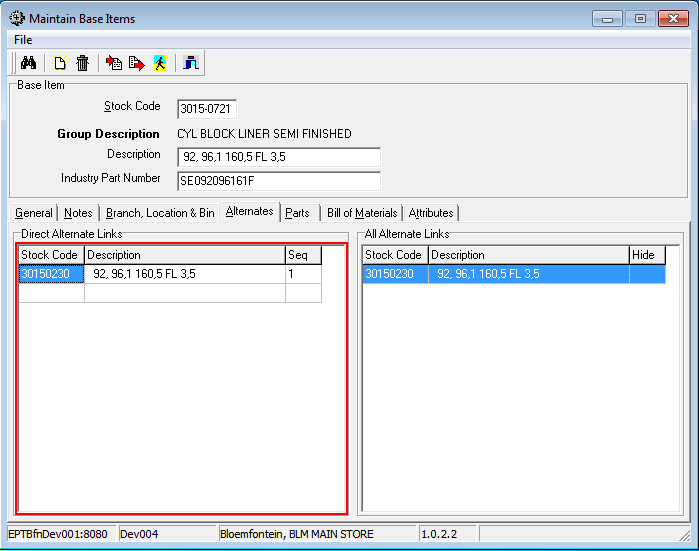
The detail is fine grained as it 1st references the Branch & location and then the bin locations where said SKU can physically be stored.

The quantities to buy-in and to sell-in maintain packaging integrity i.e. 4 pistons in a carton will force the sales person & picker to only use the stated quantity

* **Alternates**. One of the design objectives for ePart was to capture as many Alternate Part numbers as is needed. This would fit the ability to quote to customers SKU / base numbers using one or more alternates that can perform the same function; often in ePart the same roller bearing from 2 suppliers may carry differing base number due to pricing constraints.

The view below indicates the ePart inter-part linking of alternates in the following:

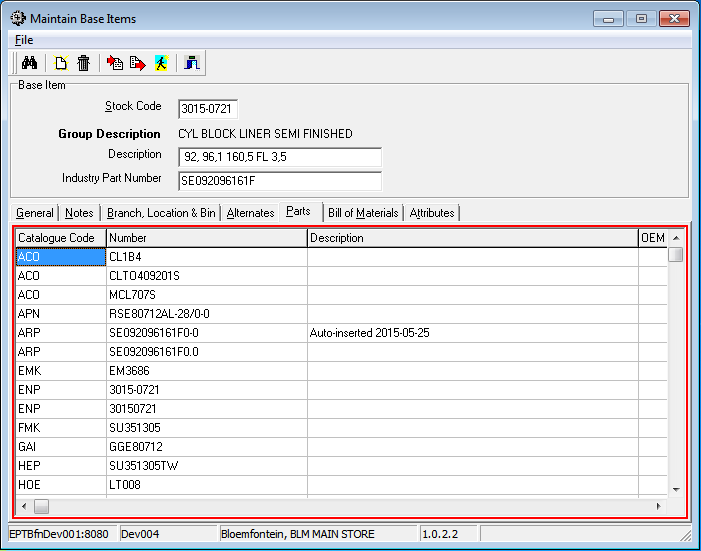
* + **Direct Alternate Links** represents alternates linked to this SKU / base number
  + **All Alternate Links** presents as a chained set of alternates.

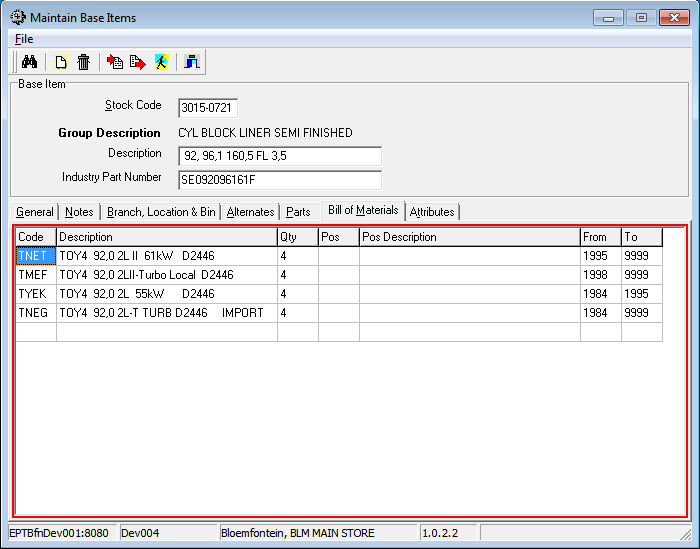


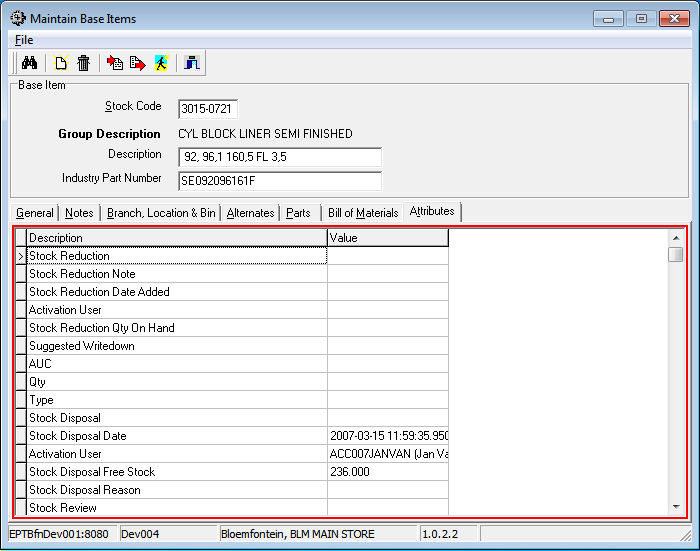
* **Parts.** These are the references to supplier catalogues where the supplier stocking code is captured and is used for looking up the ePart base number if quoted by a potential customer.

Some refinement may be possible to eliminate “-“ from the content as can be seen. This must be analysed in detail to assess if suppliers may use the “-“ as a significant character.

The OEM column signals that a specific catalogue part originates from the original manufacturer.



* **Bill of Material (Base Item)**. The design scope of ePart needed to represent all parts in a parent to child relationship so that the specific application of a part could best be understood. Additionally, this also assist the catalogue engineering team (responsible for the data maintenance) to visualise the parts application structure best.
* **Attributes**. Parts have attributes but differ from one part to the next. To best cater for the varying part profiles, the ePart engineering data maintenance team define a set of attributes per part. This set of attributes then guide the data capturers as to the data capturing requirements. The data capturing input screen auto adjust according to the attribute set where the “Description” is displayed and the “Value” that is attributed to it.



## Catalogue – Bill of material node maintenance

This part of the catalogue maintenance functionality allows for the ***Bill of Material*** nodes to be maintained.

In a generic manner the following describes why and how the BOM is defined and constructed:

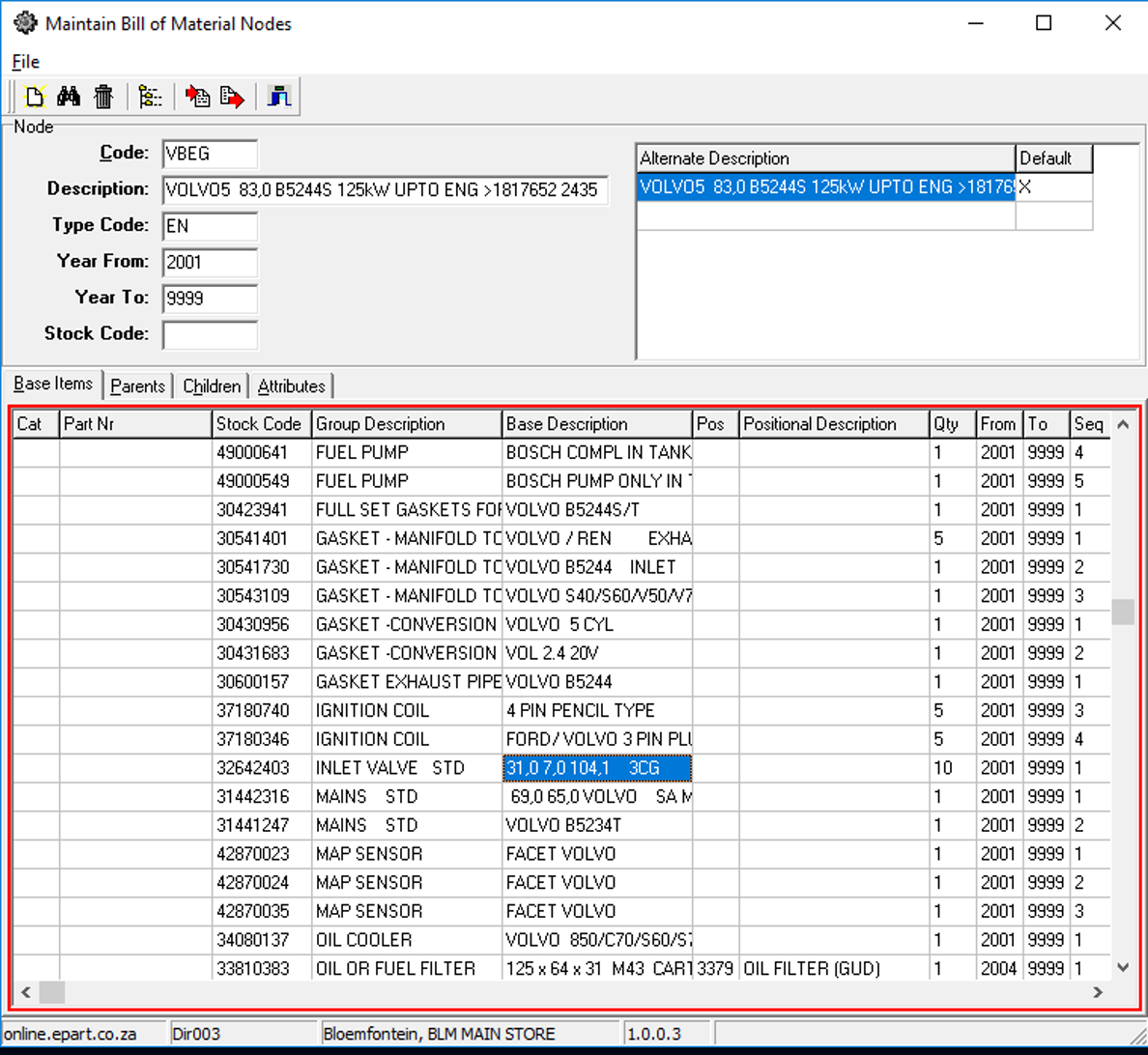
* + 1. Defining the relationship

Generically, BOM’s have a parent to child relationship and in the ePart application all relevant relationships are developed in and around automotive aftermarket interests

This implies that all automotive manufacturers and their products are defined in a specific relationship. In simplified terms it could be FORD -> ECHO SPORT -> ENGINE -> 1L -> FILTRATION ……

This allows for all relevant assemblies and sub-assemblies to be defined in a relationship that is logical and well ordered with an efficient presentation format.

Of importance is the fact that an assembly (say a turbo charger) can be sold as a stocking item and the turbo charger itself can be de-composed further into sub-assemblies (say Electrical turbo controls) and finite parts (say turbo shaft bearings) and these also sold as individual parts. This is defined by entering a ***Stock Code*** via the user interface below.



In addition to capturing the specific BOM node details, there are some tabbed detail that is used to define the BOM node in more detail.

***Tab: Base Items:***

Using this tab allows the maintainer to define all the base items (SKU) that are linked to the specific engine being defined.

Note that the engine as depicted is an assembly of type ***ENGINE***. Any other pre-defined assembly types can follow the same regimen i.e. a turbo charger or gearbox as examples fit into the same regimen of declaration. This allows for great flexibility in building the parent child relationships.

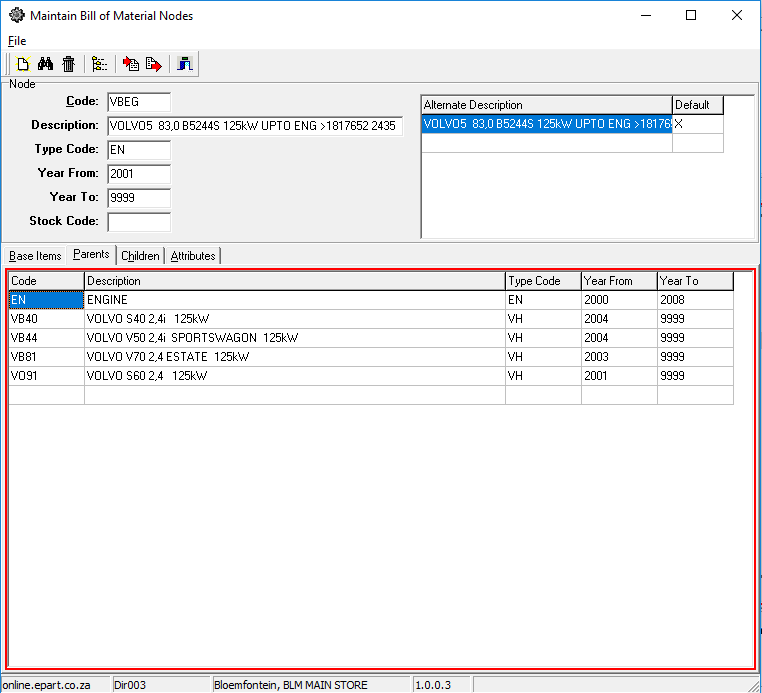
The Stock Code can be repeated in the list when the ***POs*** field is added. The functionality allows for the same part i.e. roller bearing, to be declared to fix in multiple places in the assembly.

To assist the customer and sales staff, a specific quantity is declared so that when a part is ordered there is assistance to order the correct quantity.

In the even where there is updates during the life of the assembly, a date range is provided to assist sales in selecting the correct part at enquiry time.

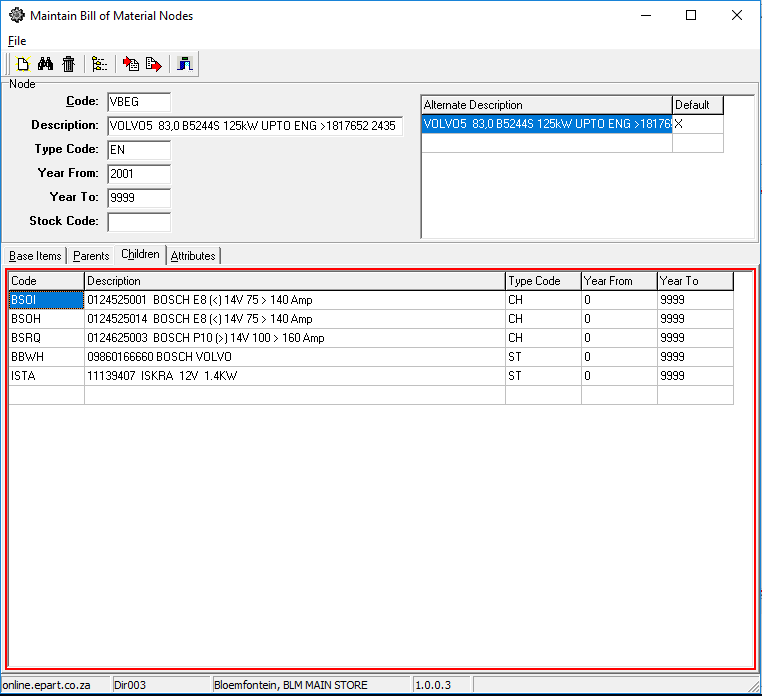
***Parents:***

Within a BOM structure, parent nodes need to be assigned to a sub-assembly. By using the ***Parents*** tab, a list of all the parents is displayed that this specific node is linked to. This regimen allows for a structure to be declared once and re-used many times over – creates efficiencies and accuracy



***Children:***

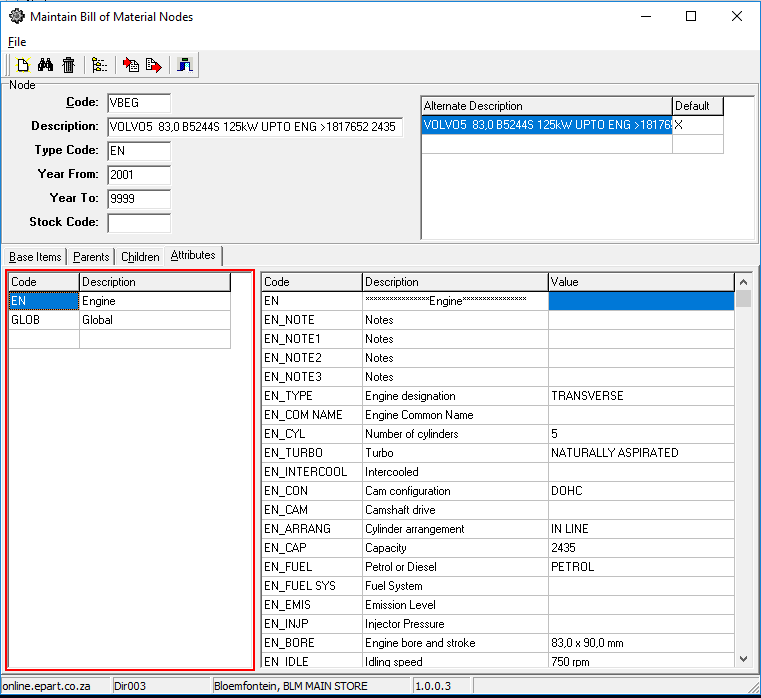
When creating an assembly, there will (possibly) be children; either in the form of sub-assemblies that make up the current assembly node.



***Attributes***

Due to the differing properties between assemblies and sub-assemblies, provision is made to define a set of attributes dynamically that allow a fixed form of elements to be presented.

In this instance the example shown depicts attributes for an ***engine*** and would differ from those for vehicle or turbo charger.



***Templates:***

In the example depicted here, and assembly of type ENGINE is being maintained.

This specific view presents all the vehicles that the specific ENGINE is linked to. This functionality is specifically designed for efficiency and accuracy in that the ENGINE assembly is pre-defined ONCE ONLY and then attached as a child to a specific vehicle.

NOTE that the ***EN*** assembly type can be defined as a ***template*** and then fully defined in its set of sub-assemblies and linked base numbers (SKU). This will automatically include that alternates as well.

By following this regimen, changes to the ***template*** that is linked to one or more parent nodes will automatically reflect modifications made to the template

## Stock Mark-up workbench

The ***Mark-up Workbench*** functionality is where end users can bring the backend of the business into a consolidated view where maintenance is done to ensure that the base calculations, pre-customer discounts, are optimally structured.

This user interface is included as it forms part of the ePart catalogue / decision support strategic configuration. From this user interface the following functions that influence the ePart catalogue efficiency can be actioned:

* Suppliers are defined as preferred suppliers to order from. This is also structured in a sequence of preference. The workbench allows the user to add to purchase order using one of the action buttons on the taskbar. The quantity is added as a **suggested** quantity to any one of the related suppliers ***open to add to*** purchase order – this is a specific supplier purchase order with status ***suggested***.

The suggested quantity is a calculation base on a formula known as the AMS (Average Monthly Sales) formula which is a rolling average of averages.

After adding the item to a purchase order in status ***suggested***, the user can change the quantity as is needed based on prevailing indicators.

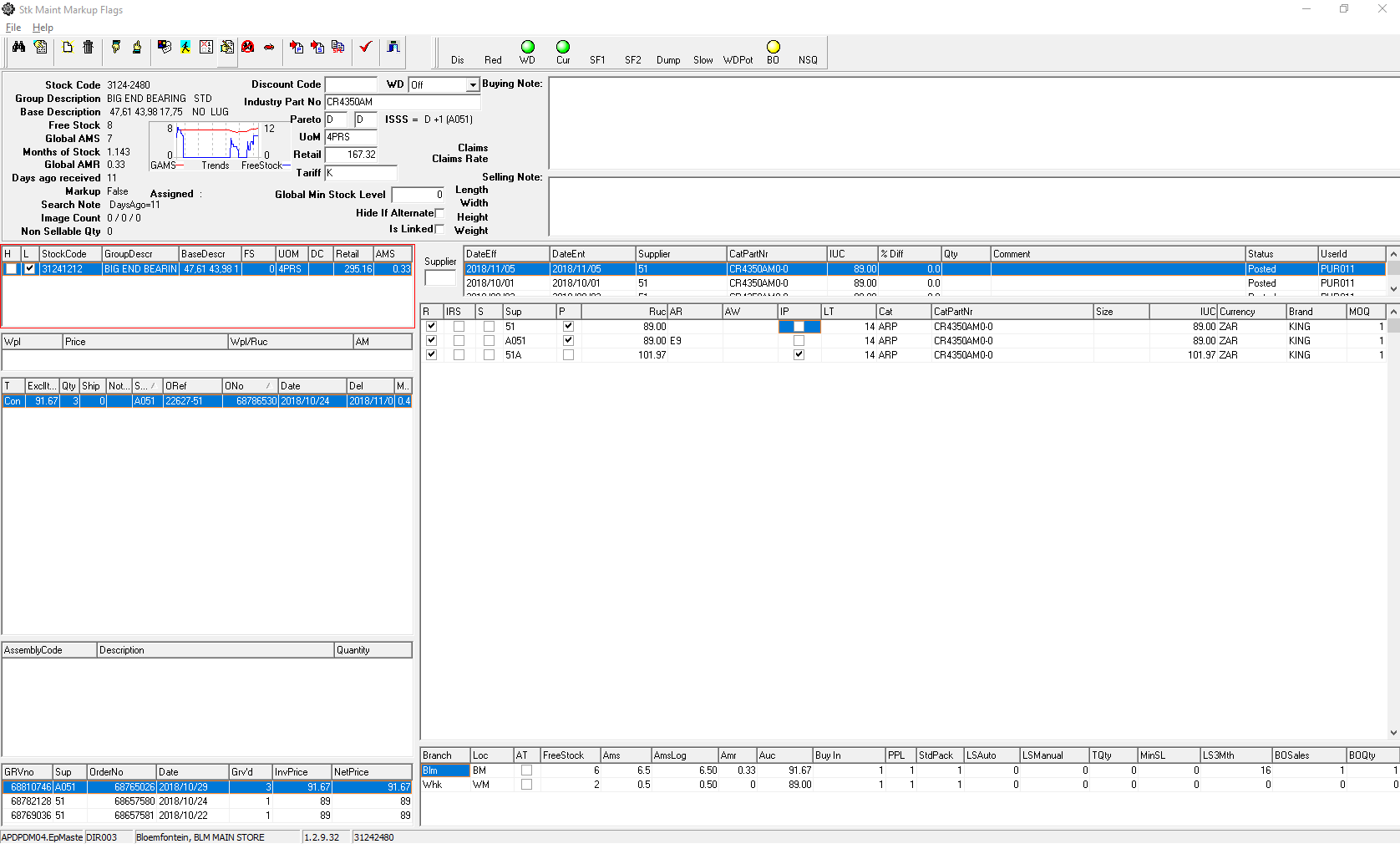
On the user view, there is statistical detail around the specific item intended to assist the user to make a more informed purchasing decision.

* Importing goods from international sources usually provide greater margins thus the preference should be to optimise imports rather than ordering from local suppliers. However, local suppliers have the advantage of receiving goods within a short time where imported goods take an appreciably longer time to receive.

Using this interface allows user to initiate the function ***add to purchase order*** amongst any number of suppliers.

* In determining selling price, suppliers can be selected for selling price determination using the supplier’s price. Distinction is made between normal as well as whole sale selling price determination.

Technical: To optimise the price calculation process, there is a database trigger that is initiated to do pre-calculation of the base selling pricing lists using the revised supplier selection and their price to Engineparts.

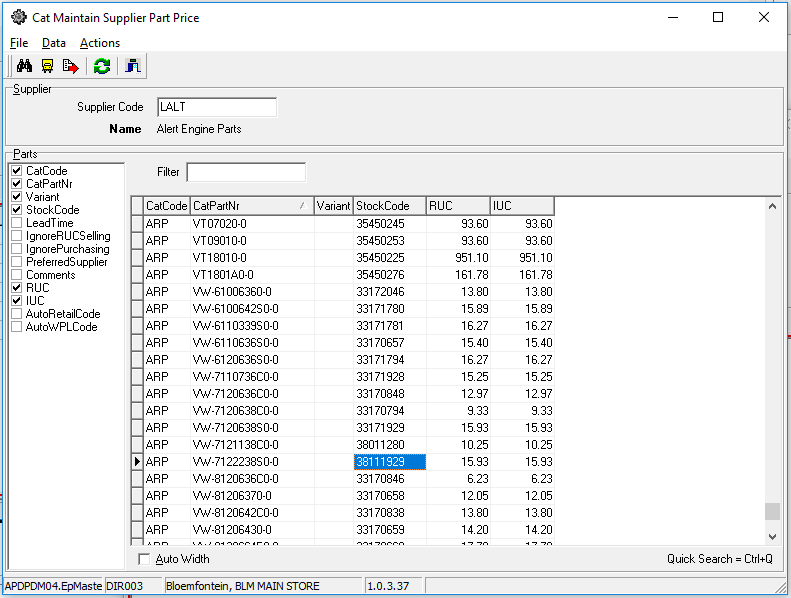
 A monthly trend graph is available for monthly sales to assist in defining the optimal pricing constructs.

## Catalogue Supplier Price Maintenance

In addition to the automated price maintenance, there is a functionality to maintain the relationship between catalogues, suppliers, catalogue part numbers and to maintain specific values in the relationships.

The purpose of this is to maintain the relationships in support of the strategic model in support of optimised purchasing and selling price determination.

Depending on a user’s business focus, this user view can be modified dynamically by using the set of tick-boxes in the ***parts*** display section. By setting a tick option on adds the specific column to the right-hand display section. For instance, if the user is tasked to review and update the ***Lead Time*** then the Lead time tick box is ticked allowing the said field to be edited.

From the display it is possible to see which columns in the display area is ticked to be viewed and edited.

# Dependencies

|  |  |  |
| --- | --- | --- |
| # | Description | Action / By whom |
| 1 | Accounts receivable |  |
| 2 | Pricing |  |
| 3 | Catalogue maintenance |  |
| 4 | Discount structures |  |
| 5 | Warehouse activities |  |
| 6 | Sales order |  |

# Database entities and relationships

The full catalogue related database diagram provides a view of the various participating relationships. Notably is the simplicity of the database tables participating in the core structure.



The catalogue support system provides for a highly efficient source of information, provided the underlying data is accurately maintained and kept up to date; failing which will result in poor customer experience.

# Programs

# MS Windows Executables

|  |  |
| --- | --- |
| **Name** | **Description** |
| CatBookWorkbench |  |
| CatMaintAttribute |  |
| CatMaintBaseItem |  |
| CatMaintBaseItemMeasure |  |
| CatMaintBomNode |  |
| CatMaintBomNodeType |  |
| CatMaintBook | Maintains all supplier catalogue materials as a library. |
| CatMaintCatalogue |  |
| CatMaintGroup |  |
| catMaintSupplierPartPrice |  |

# SQL Stored Procedures

The following stored procedures use a standardized naming convention and an object-oriented paradigm.

* Starting with “cat” indicates the cataloguing subsystem.
* The end of the name indicates the purpose:
  + HomeCreate - Inserts a new object record.
  + HomeDestroy - Deletes an object record.
  + HomeFindX - Find an object by field X.
  + FieldChange - Updates and validates the data in the object.
  + Refresh - Retrieves the data record of the object.
* The part between “cat” and the ending indicates the object class.
* Further detail can be read from the code in the SQL files.
* At the end of each SQL file there is also a test harness in a block comment, to be run after each change, for verification.

|  |
| --- |
| catAlternateFieldChange.sql |
| catAlternateHomeCreate.sql |
| catAlternateHomeDestroy.sql |
| catAlternateHomeFindByStockCode.sql |
| catAlternateRefresh.sql |
| catAssemblyHomeFindByQueryNo.sql |
| catAssemblyRefresh.sql |
| catAttBomNodeFieldChange.sql |
| catAttBomNodeHomeFindByBomNodeCodeAttGroupCode.sql |
| catAttBomNodeRefresh.sql |
| catAttGroupFieldChange.sql |
| catAttGroupHomeCreate.sql |
| catAttGroupHomeDestroy.sql |
| catAttGroupHomeFindAll.sql |
| catAttGroupHomeFindByBomNodeCode.sql |
| catAttGroupItemFieldChange.sql |
| catAttGroupItemHomeCreate.sql |
| catAttGroupItemHomeDestroy.sql |
| catAttGroupItemHomeFindByAttGroupCode.sql |
| catAttGroupItemRefresh.sql |
| catAttGroupRefresh.sql |
| catAttributeFieldChange.sql |
| catAttributeHomeCreate.sql |
| catAttributeHomeDestroy.sql |
| catAttributeHomeFindAll.sql |
| catAttributeRefresh.sql |
| catBaseItemAttributeChangeField.sql |
| catBaseItemAttributeHomeFindByStockCode.sql |
| catBaseItemAttributeHomeFindByStockCodePrefix.sql |
| catBaseItemAttributeRefresh.sql |
| catBaseItemAttributeRefreshDynamic.sql |
| catBaseItemFieldChange.sql |
| catBaseItemHomeCreate.sql |
| catBaseItemHomeFindAlternates.sql |
| catBaseItemHomeFindByCatCodePartNr.sql |
| catBaseItemHomeFindVariants.sql |
| catBaseItemLinkHomeFindByStockCode.sql |
| catBaseItemLinkRefresh.sql |
| catBaseItemNoteRefresh.sql |
| catBaseItemRefresh.sql |
| catBomBaseItemFieldChange.sql |
| catBomBaseItemHideUnder.sql |
| catBomBaseItemHomeCreate.sql |
| catBomBaseItemHomeDestroy.sql |
| catBomBaseItemHomeFindByBomNodeCode.sql |
| catBomBaseItemHomeFindByStockCode.sql |
| catBomBaseItemIsHiddenUnder.sql |
| catBomBaseItemRefresh.sql |
| catBomBaseItemShowUnder.sql |
| catBomNodeAddLink.sql |
| catBomNodeAltDescrFieldChange.sql |
| catBomNodeAltDescrHomeCreate.sql |
| catBomNodeAltDescrHomeDestroy.sql |
| catBomNodeAltDescrHomeFindByCode.sql |
| catBomNodeAltDescrRefresh.sql |
| catBomNodeBuildBridge.sql |
| catBomNodeCheckForLoop.sql |
| catBomNodeFieldChange.sql |
| catBomNodeHomeCreate.sql |
| catBomNodeHomeDestroy.sql |
| catBomNodeHomeFindByStockCode.sql |
| catBomNodeHomeFindChildren.sql |
| catBomNodeHomeFindParents.sql |
| catBomNodeHomeFindRoot.sql |
| catBomNodeHomeGetNewCode.sql |
| catBomNodeRefresh.sql |
| catBomNodeRemoveLink.sql |
| catBomNodeTypeFieldChange.sql |
| catBomNodeTypeHomeCreate.sql |
| catBomNodeTypeHomeDestroy.sql |
| catBomNodeTypeHomeFindAll.sql |
| catBomNodeTypeRefresh.sql |
| catBookFieldChange.sql |
| catBookHomeCreate.sql |
| catBookHomeDestroy.sql |
| catBookKeywordFieldChange.sql |
| catBookKeywordHomeCreate.sql |
| catBookKeywordHomeDestroy.sql |
| catBookKeywordHomeFindByBookId.sql |
| catBookKeywordRefresh.sql |
| catBookPageFieldChange.sql |
| catBookPageHomeCreate.sql |
| catBookPageHomeDestroy.sql |
| catBookPageHomeFindByBookId.sql |
| catBookPageImageDetail.sql |
| catBookPageNoteFieldChange.sql |
| catBookPageNoteHomeCreate.sql |
| catBookPageNoteHomeDestroy.sql |
| catBookPageNoteHomeFindByBookIdPageId.sql |
| catBookPageNoteHomeFindByPagePartId.sql |
| catBookPageNoteRefresh.sql |
| catBookPageRefresh.sql |
| catBookRefresh.sql |
| catCatalogueFieldChange.sql |
| catCatalogueHomeCreate.sql |
| catCatalogueHomeDestroy.sql |
| catCatalogueRefresh.sql |
| catCatQueryBomNodeHomeFindByQueryNo.sql |
| catCatQueryFieldChange.sql |
| catCatQueryHomeCreate.sql |
| catCatQueryHomeFindLast.sql |
| catCatQueryHomeFindNext.sql |
| catCatQueryHomeFindPrior.sql |
| catCatQueryRefresh.sql |
| catCatQueryResultHomeFindByQueryNo.sql |
| catCatQueryResultHomeFindByQueryNo2.sql |
| catGroupAddAttributeGroup.sql |
| catGroupAltDescrFieldChange.sql |
| catGroupAltDescrHomeCreate.sql |
| catGroupAltDescrHomeDestroy.sql |
| catGroupAltDescrHomeFindByCode.sql |
| catGroupAltDescrRefresh.sql |
| catGroupFieldChange.sql |
| catGroupHomeCreate.sql |
| catGroupHomeDestroy.sql |
| catGroupHomeFindByBaseCode.sql |
| catGroupHomeFindByStdSizeCode.sql |
| catGroupRefresh.sql |
| catGroupRemoveAttributeGroup.sql |
| catGroupViewAttribGroups.sql |
| catPagePartAltFieldChange.sql |
| catPagePartAltHomeCreate.sql |
| catPagePartAltHomeDestroy.sql |
| catPagePartAltHomeFindByPagePartId.sql |
| catPagePartAltRefresh.sql |
| catPagePartFieldChange.sql |
| catPagePartHomeCreate.sql |
| catPagePartHomeDestroy.sql |
| catPagePartHomeFindByBomNodeCode.sql |
| catPagePartHomeFindByBookIdPageId.sql |
| catPagePartHomeFindByStockCode.sql |
| catPagePartImageDetail.sql |
| catPagePartRefresh.sql |
| catPartFieldChange.sql |
| catPartHomeCreate.sql |
| catPartHomeDestroy.sql |
| catPartHomeFindByStockCode.sql |
| catPartRefresh.sql |
| catProgressPartListAddLine.sql |
| catProgressPartListCreate.sql |
| catSetGroupFieldChange.sql |
| catSetGroupHomeCreate.sql |
| catSetGroupHomeDestroy.sql |
| catSetGroupHomeFindBySetCode.sql |
| catSetGroupRefresh.sql |
| catSupplierPartMove.sql |
| catSupplierPartSplit.sql |

# Acceptance

I hereby confirm that I have been fully informed of the document’s content and received training to understand how the detailed instructions are to be applied:

Name …………………………………………………………………………….

Job Title ………………………………………………………………………….

Signed ……………………………………………………………………………

Date ………………………………………………………………………………